Rsync:

Tutorial and Examples to Copy, Mirror, Synchronize, Backup and Restore files

Rsync is a mechanism to mirror data on a single host or between multiple hosts.
Rsync can duplicate directories and files and provide backup/restore operations or mobile device synchronization. This tutorial covers Rsync client and server configurations for both Linux and MS/Windows (using Cygwin).

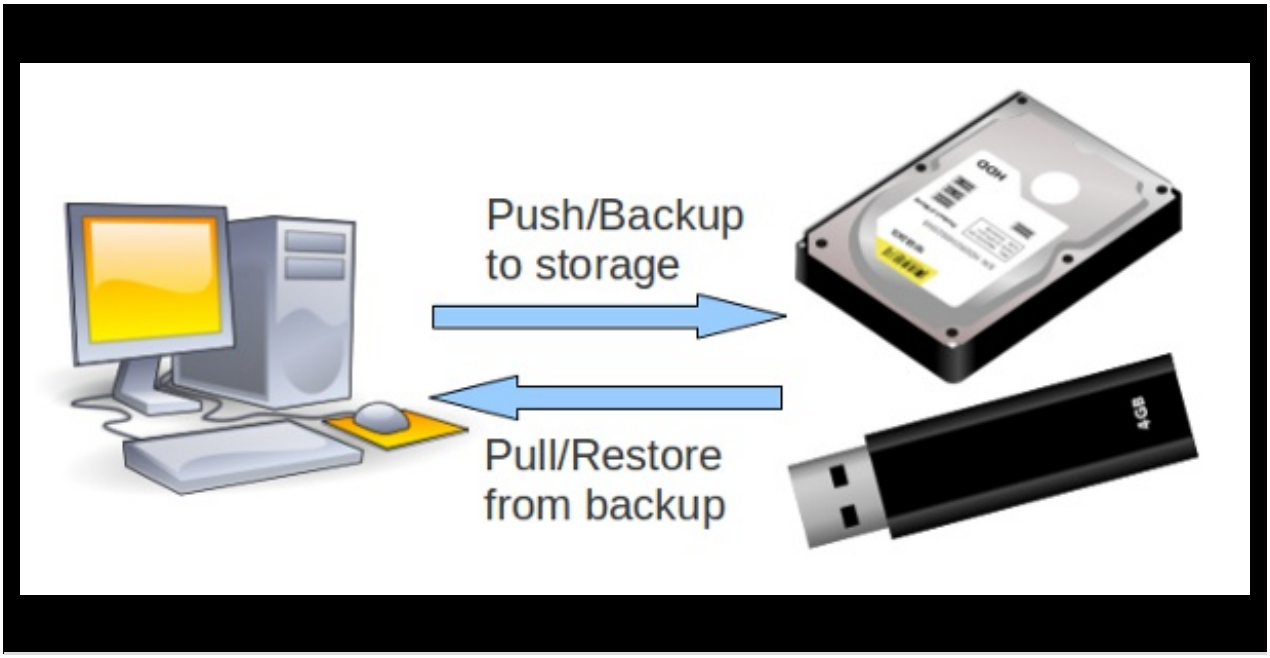
Rsync description, topologies and configuration:

Rsync can duplicate data between two directories whether the data is collocated on the same computer or over the network between two computers. The directories can be on any accessible random access storage devices (hard drives, thumb drives, etc). The data source must be readable and the destination must be writable.

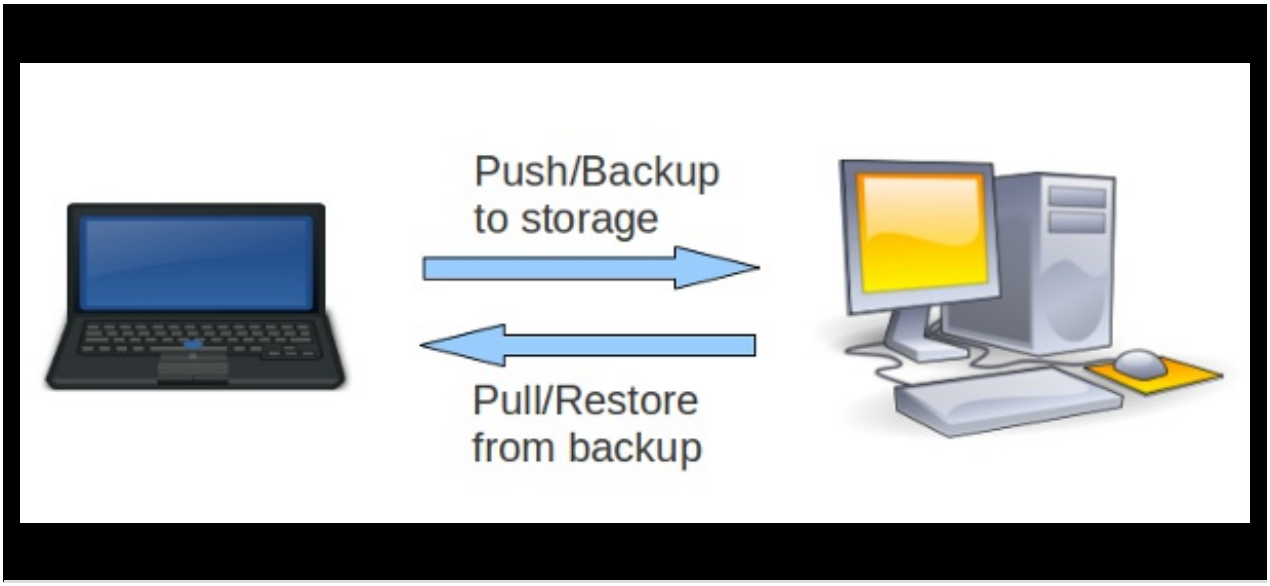
The benefit of Rsync is the ability to update the mirrored backup with computed changes from the data source with a minimal load to system resources. It is very data bandwidth and system intensive to perform an entire copy of a directory structure each time a backup is to be performed. While the initial backup is no faster than a copy, Rsync has the ability to load the system in a minimal fashion by transferring only changes when subsequent rsync updates are performed.

Description of the rsync algorithm

Rsync single host: mirror between storage devices



Rsync client-server: mirror between two computers



Rsync Single Host Operation:

Rsync command format: `rsync [options] source-path destination-path`

Synchronize directory paths each of which are accessible to a single system.

Examples:

- Back-up (push) your photo directories to a second drive:
  `rsync -av ~/Pictures /mnt/drive2`
  This creates a backup of your photos in `/mnt/drive2/Pictures/`
  Back-up to a USB thumb drive is similar: `rsync -av ~/Pictures /media/KINGSTON`
  When you add new photos, just re-execute this rsync command to backup the latest changes.
  Note: The drive name will be dependent on the manufacturer.

  [Potential Pitfall]: Do not include the source directory name in the destination.
  `rsync -av ~/Pictures /mnt/drive2/Pictures`
  This will result in `/mnt/drive2/Pictures/Pictures/`
  Note that rsync destinations acts just like the cp and rcp commands.
  Also note that `rsync -av ~/Pictures/ /mnt/drive2/` has a different behavior from `rsync -av ~/Pictures /mnt/drive2`

- Back-up (push) two directory source to a single directory on the second drive:
  `rsync -av ~/Pictures ~/Images /mnt/drive2`
  This creates a backup of your photos from the two directories in `/mnt/drive2/Pictures/`

- Sync directories and if any were deleted in `~/Pictures`, also delete it in `/mnt/drive2/Pictures/`:
  `rsync -a --progress --delete ~/Pictures /mnt/drive2`
  This creates a backup in `/mnt/drive2/Pictures/`

- Sync one specific file in a directory:
  `rsync -a ~/Pictures/Group-photo-2001-A.jpg /mnt/drive2/Pictures`

- Sync a group of specific files in a directory:
  `rsync -a ~/Pictures/2001-*.jpg /mnt/drive2/Pictures`
  This creates a backup in `/mnt/drive2/Pictures/`
  Note that when transferring files only, the directory name has to be provided in the destination path.

- Sync files and directories listed in a file:
  `rsync -ar --files-from=Filelist.txt ~/Data /mnt/drive2`
  This creates a backup in `/mnt/drive2/Data/`
  Directory paths are included if specified with a closing slash such as pathx/pathy/pathz/. Path names must be terminated with a "/" or "/."

- Back-up (push) your source code and compress to save space. Ignore object files:
  `rsync -avz --delete --exclude='*.o' --exclude='*.so' ~/src /mnt/drive2`
  This creates a backup in `/mnt/drive2/src/` but does not transfer files with the ".o" and ".so" extensions.

- Back-up (push) your source code and ignore object and shared object code files:
  `rsync -av --delete --filter='+ *.[ch]' --filter='- *.o' ~/src /mnt/drive2`
  This transfers files with the extension ".c" and ".h" but does not transfer object files with the ".o" extensions.
  same as `rsync.exe -av --exclude='*.o' --filter='+ *.c *.h' ~/src /mnt/drive2`

- Back-up (push) your source code and ignore CM directories, object and shared object code files:
  `rsync -artv --progress --delete --filter='+ *.[ch]' --filter='- *.o' --exclude=".svn" ~/src /mnt/drive2`
  Note that `--exclude` overides the include filter `--filter='+ *.[ch]'` so that ".c" and ".h" files under `.svn/` are not copied.

Referencing directories - errors and subtleties:

- `rsync -ar dir1 dir2`
  This will copy `dir1` into `dir2` to give `dir2/dir1` thus `dir1/file.txt` gets copied to `dir2/dir1/file.txt`

- `rsync -ar dir1/ dir2/`
  This will copy the contents of `dir1` into `dir2` to give `dir2/contents-of-dir1`, eg `dir1/file.txt` gets copied to `dir2/file.txt`
  The following all achieve the same results:
  - `rsync -ar dir1/ dir2/`
  - `rsync -ar dir1/* dir2`
  - `rsync -ar dir1/* dir2/`

Rsync Options: (partial list)

Command line argument	Description
-a (--archive)	Archive. Includes options: • -r: recursion • -l: preserve symbolic links as symbolic links. Opposite of -L • -p: preserve permissions (Linux/unix only) • -t: preserve file modification time • -g: preserve group ownership • -o: preserve user ownership • -D: preserve special files and devices (Linux/unix only)
-d (--dirs)	Copy directory tree structure without copying the files within the directories
--existing	Update only existing files from source directory which are already present at the destination. No new files will be transferred.
-L (--copy-links)	Transform a symbolic link to a copied file upon transfer
--stats	Print verbose set of statistics on the transfer Add `-h` (`--human-readable`) to print stats in an understandable fashion
-p (--perms)	Preserve permissions (not relevant for MS/Windows client)
-r (--recursive)	Recursive through directories and sub-directories
-t (--times)	Preserve file modification times
-v (--verbose)	Verbose

-z (--compress)	Compress files during transfer to reduce network bandwidth. Files not stored in an altered or compressed state. Note that compression will have little or no effect on JPG, PNG and files already using compression. Use arguments `--skip-compress=gz/bz2/jpg/jpeg/ogg/mp[34]/mov/avi/rpm/deb/` to avoid compressing files already compressed
--delete	Delete extraneous files from destination directories. Delete files on archive server if they were also deleted on client. Use the argument `-m` (`--prune-empty-dirs`) to delete empty directories (no longer useful after its contents are deleted)
--include --exclude --filter	Specify a pattern for specific inclusion or exclusion or use the more universal filter for inclusion (+)/exclusion (-). Do not transfer files ending with ".o": `--exclude='*.o'` Transfer all files ending with ".c" or ".h": `--filter='+ *.[ch]'`
-i (--itemize-changes)	Print information about the transfer. List everything (all file copies and file changes) rsync is going to perform
--list-only --dry-run	Don't copy anything, just list what rsync would copy if this option was not given. This helps when debugging the correct exclusion/inclusion filters.
--progress	Shows percent complete, Kb transferred and Kb/s transfer rate. Includes verbose output.

For all options see the rsync man page

Note that rsync will be able to handle files with blanks in the file name or directory name as well as with dashes ("-") or underscores ("_").

Rsync Client-Server Configuration and Operation:

Rsync can be configured in multiple client-server modes.

1. connect client to a sever running rsync in daemon mode
2. connect client to a sever using a ssh shell

These configurations are specified with the use of the colon ":"

- Double colon refers to a connection to a host running the rsync daemon in the format hostname::module/path where the module name is identified by the configuration in /etc/rsyncd.conf. The double colon is equivalent to using the URL prefix `rsync://`
- Single colon refers to the use of a remote shell
- No colon then the directory is considered to be local to the system.

---

1) Rsync daemon server:

The Rsync server is often referred to as rsyncd or the rsync daemon. This is in fact the same rsync executable run with the command line argument "`--daemon`". This can be run stand-alone or using xinetd as is typically configured on most Linux distributions.

Configure xinetd to manage rsync:

File: `/etc/xinetd.d/rsync`

Default: "`disable = yes`". Change to "`disable = no`"

```
01  service rsync
02  {
03          disable =
            no
04          flags           =
            IPv6
05          socket_type     =
            stream
06          wait            =
            no
07          user            =
            root
08          server          =
            /usr/bin/rsync
09          server_args     = --
            daemon
10          log_on_failure  +=
            USERID
11  }
```

Start/Re-start xinetd: `/etc/rc.d/init.d/xinetd restart`

For more information on xinetd see the YoLinux xinetd tutorial.

Typical Linux distributions do not pre-configure rsync for server use. Both Ubuntu and Red Hat based distributions require that one generates the configuration file "`/etc/rsyncd.conf`"

File: `/etc/rsyncd.conf`

```
01  log file =
    /var/log/rsyncd.log

02  hosts allow = 192.17.39.244,
    192.17.39.60

03  hosts deny =
    *

04  list =
    true

05  uid =
    root

06  gid =
    root

07  read only =
    false

08

09  [Proj1]

10  path =
    /tmp/Proj1

11  comment = Project 1 rsync
    directory

12

13  [ProjX]

14  path =
    /var/ProjX

15  comment = Project X rsync
    directory
```

rsyncd.conf man page

Client command to rsync to the server:

Push: `rsync -avr /home/user1/Proj1/Data server-host-name::Proj1`
(eg. update server backup from mobile laptop)
This will initially copy over directory Data and all of its contents to `/tmp/Proj1/Data` on the remote server.

Pull: `rsync -avr server-host-name::Proj1 /home/user1/Proj1/Data`
(eg. update mobile laptop from server backup)

---

2) Rsync to server using ssh shell:

Using this method does not use the configuration "modules" in `/etc/rsyncd.conf` but instead uses the paths as if logged in using ssh.

First configure ssh for "password-less" login:
Note that current Linux distributions use ssh version 2 and rsa.

```
[user1@myclient ~]$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/user1/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/user1/.ssh/id_rsa.
Your public key has been saved in /home/user1/.ssh/id_rsa.pub.
The key fingerprint is:
aa:1c:76:33:8a:9c:10:51:.....
```

Note that "Enter" was pressed when asked for a "passphrase" to take the default.
Two files are generated:

- Local client (private key): ~/.ssh/id_rsa
- Contents (one line) of file (public key): ~/.ssh/id_rsa.pub to be copied into file on server: ~/.ssh/authorized_keys

Note file protection on file:

```
[user1@myclient ~]$ ls -l ~/.ssh/id_rsa
-rw-------. 1 user1 user1 1675 Sep  7 14:55
/home/user1/.ssh/id_rsa
```

Copy public key to server so you can login.
Use the following command: Now try logging into the machine, with "ssh 'user1@remote-server'", and check in: .ssh/authorized_keys to make sure we haven't added extra keys that you weren't expecting.

Test "password-less" ssh connection: `ssh remote-server`
This command should log you in without asking for a password.

Now try rsync (push) using ssh:

```
rsync -avr --rsh=/usr/bin/ssh /home/user1/Proj1/Data remote-
server:/mnt/supersan/Proj1
```

Note that if this connection is is to be spawned by a cron job (eg. root user) then the shell user ID must be provided: `user1@`

```
rsync -avr --rsh=/usr/bin/ssh /home/user1/Proj1/Data user1@remote-
server:/mnt/supersan/Proj1
```

SSH options may be put in the file `~/.ssh/config`

---

crontab:

Note that rsync is often used with cron to perform a nightly rsync.

eg. Rsync to get latest updates to the web server at 2:00am:

File: `/etc/crontab`

```
1  * 2 * * * rsync -avr server-host-name::Proj1/html /var/www >
   /var/log/rsync 2>&1
```

See the crontab man page

MS/Windows Rsync Client and Server Operation:

Multiple solutions exist for MS/Windows rsync clients but they all rely on Cygwin. There is no other port of rsync.

- Cygwin: Work in Cygwin shell environment
- DeltaCopy (client and server)
  Uses Rsync algorithm and protocol
- QtdSync: (client and server) MS/Windows version includes Cygwin rsync and ssh (and more)
  [download]

Cygwin rsync client for MS/Windows:

Cygwin must be installed for the purpose of this tutorial.
Be sure to install the Cygwin component "Net" + "rsync".

Rsync Cygwin components:

```
  C:\cygwin\bin\rsync.exe
               \ssh.exe
               \sshpass.exe
               \cygcrypto-
0.9.8.dll
               \cyggcc_s-1.dll
               \cygiconv-2.dll
               \cygpopt-0.dll
               \cygssp-0.dll
               \cygwin1.dll
               \cygz.dll
```

These components can be extracted for a minimal installation of be left in place as part of the complete Cygwin package.

DOS bat script to perform an rsync (push):

```
1 http://twitter.com/set">@set
  PATH=C:\cygwin\bin
```

```
2 http://twitter.com/set">@set
  LIB=C:\cygwin\bin
```

```
3 rsync.exe -avr /cygdrive/c/MISC/JUNK  remote-
  server::Proj1
```

[Potential Pitfall]: The following command: `rsync.exe -avr c:MISC\JUNK remote-server::Proj1` will produce an error because rsync will interpret the "C:" as the remote host and not a drive letter. Drive letters are not POSIX compliant.

[Potential Pitfall]: The following commands:

```
cd c:\
rsync.exe -avr MISC\JUNK  remote-
server::Proj1
```

will generate a single directory of the name 'MISC\JUNK' (not two directories, one within the other but just one with that name). The "/" will not be a directory delimiter but a character in the directory name (bad).

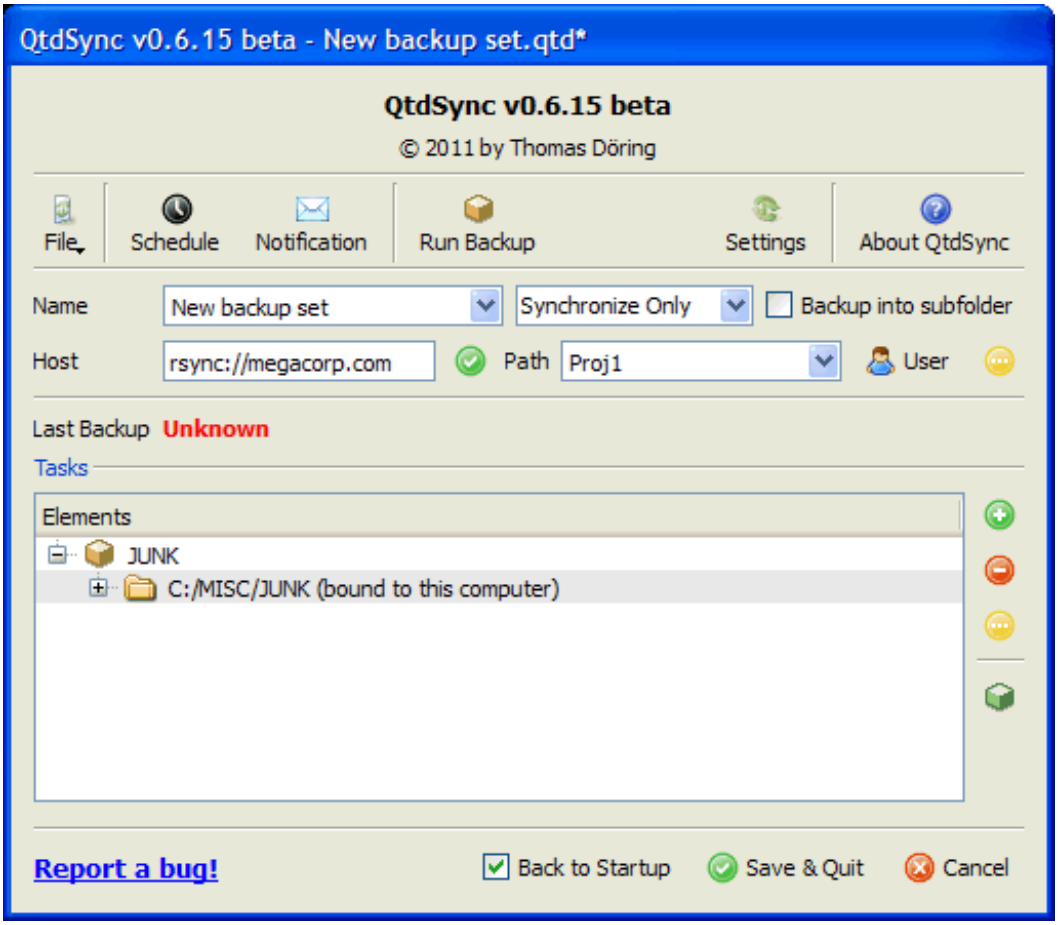
For more on Cygwin POSIX paths: http://cygwin.com/cygwin-ug-net/using.html#using-pathnames

---

MS/Windows rsync client GUI:

I chose QtdSync/cygwin as it provided a GUI and an isolated Cygwin rsync executable and its dependencies.

QtdSync GUI:



To list GUI command line arguments: `C:\Program Files\QtdSync\QtdSync.exe --help`

Includes rsync Cygwin components:

```
  C:\Program Files\QtdSync\bin\rsync.exe
                              \ssh.exe
                              \sshpass.exe
                              \cygcrypto-
0.9.8.dll
                              \cyggcc_s-1.dll
                              \cygiconv-2.dll
                              \cgypopt-0.dll
                              \cygssp-0.dll
                              \cygwin1.dll
                              \cygz.dll
                          \etc\fstab
```

Where the file `C:\Program Files\QtdSync\etc\fstab` has the following contents:

```
none /cygdrive cygdrive binary,posix=0,user,noacl 0
0
```

Note: QtdSync does not require the separate installation of cygwin.

DOS bat script to perform an rsync (push):

```
1  http://twitter.com/set">@set PATH=C:\Program
   Files\QtdSync\bin
```

```
2  http://twitter.com/set">@set LIB=C:\Program
   Files\QtdSync\bin
```

```
3  rsync.exe -avr /cygdrive/c/MISC/JUNK  remote-
   server::Proj1
```

A Cygwin MS/Windows rsyncd server configuration:

Rsync daemon configuration requires installing rsync as a MS/Windows service. Use the Cygwin command `cygrunsrv` command (installed under the Cygwin "Admin" category).

cygrunsrv command options:

Option	Description
-I --install	Install as a service
-R --remove	Remove as a service
-S --start	Start the service
-E --stop	Stop the service
-a --args	Arguments to pass the daemon service
-h	Command help instructions

Install rsync as a MS/Windows service: `regrsyncd.bat`

```
1  http://twitter.com/set">@set
   PATH=C:\cygwin\bin
```

```
2  http://twitter.com/set">@set
   LIB=C:\cygwin\bin
```

```
3  cygrunsrv -I rsyncd -e CYGWIN=nontsec --path
   /cygdrive/c/cygwin/bin/rsync.exe \
```

```
4            -a "--daemon --config=/etc/rsyncd.conf --no-
             detach"
```

You may be required to include the system user login and password:

```
cygrunsrv -I rsyncd -e CYGWIN=nontsec --path /cygdrive/c/cygwin/bin/rsync.exe
\
          -a "--daemon --config=/etc/rsyncd.conf --no-detach" -u
Administrator -w super-secret-password
```

Note that Cygwin applications refer to configuration file referenced within the Cygwin environment. As viewed by the Cygwin rsync application, the file `/etc/rsyncd.conf` is actually `C:\cygwin\etc\rsyncd.conf` as viewed by the MS/Windows OS.

Where the option `-e CYGWIN=nontsec` turns off Microsoft Windows NT security permissions. On MS/Windows 7 use the following `c:\cygwin\etc\fstab` settings:

```
1  none /cygdrive cygdrive binary,posix=0,user,noacl 0
   0
```

fstab man page

This is to avoid the following error:
`rsync: failed to modify permissions on xxxxfilenamexxxx: Permission denied`

File: `C:\cygwin\etc\rsyncd.conf`

```
01  log file =
    /var/log/rsyncd

02  hosts allow = 192.17.39.244,
    192.17.39.60

03  hosts deny =
    *

04  list =
    true

05  use chroot =
    false

06  strict modes =
    false

07  read only =
    false

08  ignore nonreadable =
    yes

09  dont compress = *.gz *.tgz *.zip *.rpm *.deb *.iso *.bz2 *.jpg *.mpg
    *.mpeg

10

11  [Proj1]

12  path =
    /cygdrive/c/Proj1

13  comment = Project 1 rsync
    directory
```

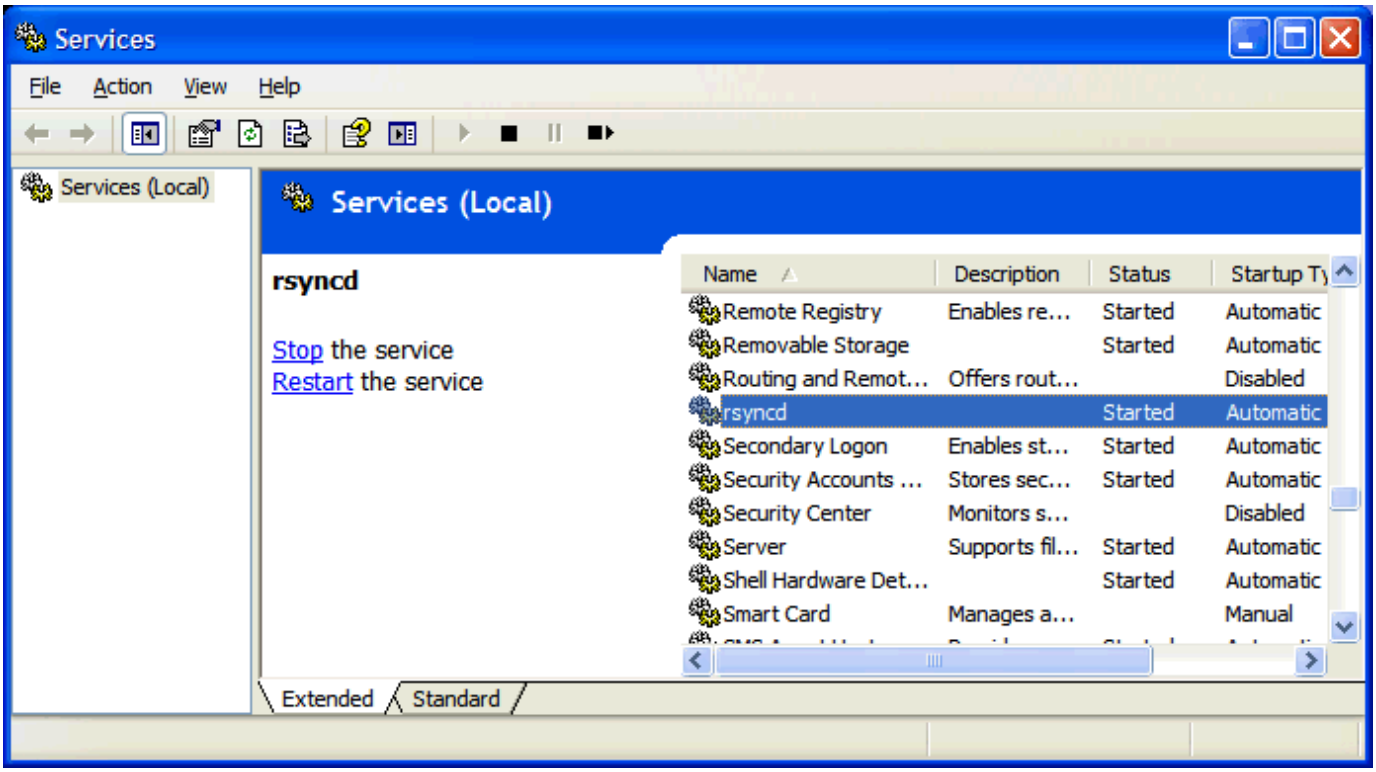
---

rsyncd.conf man page

Where the Cygwin log file `/var/log/rsyncd` is `C:\cygwin\var\log\rsyncd` as viewed by the MS/Windows OS.

The Cygwin rsync directory path `/cygdrive/c/Proj1` is `C:\Proj1\` as viewed by the MS/Windows OS.

Check to see if the rsyncd service is running: `Control Panel --> Administrative Tools --> Services`

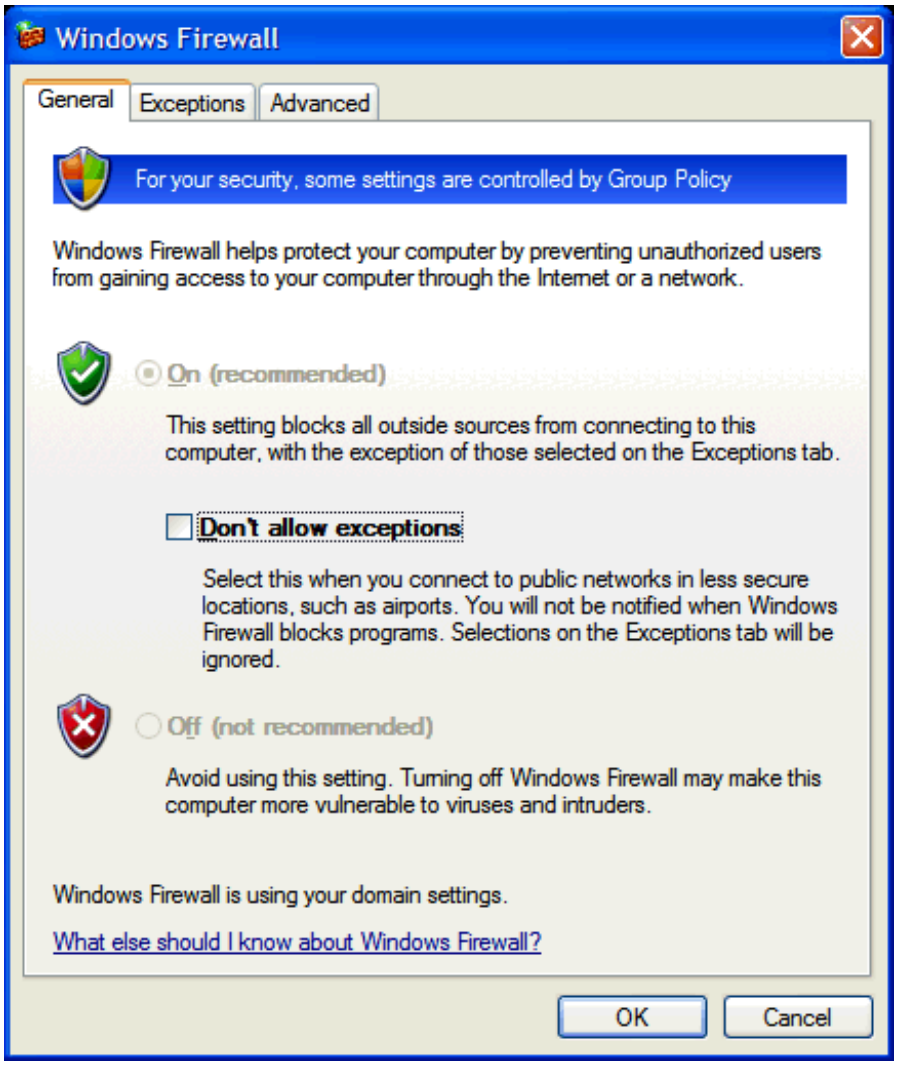


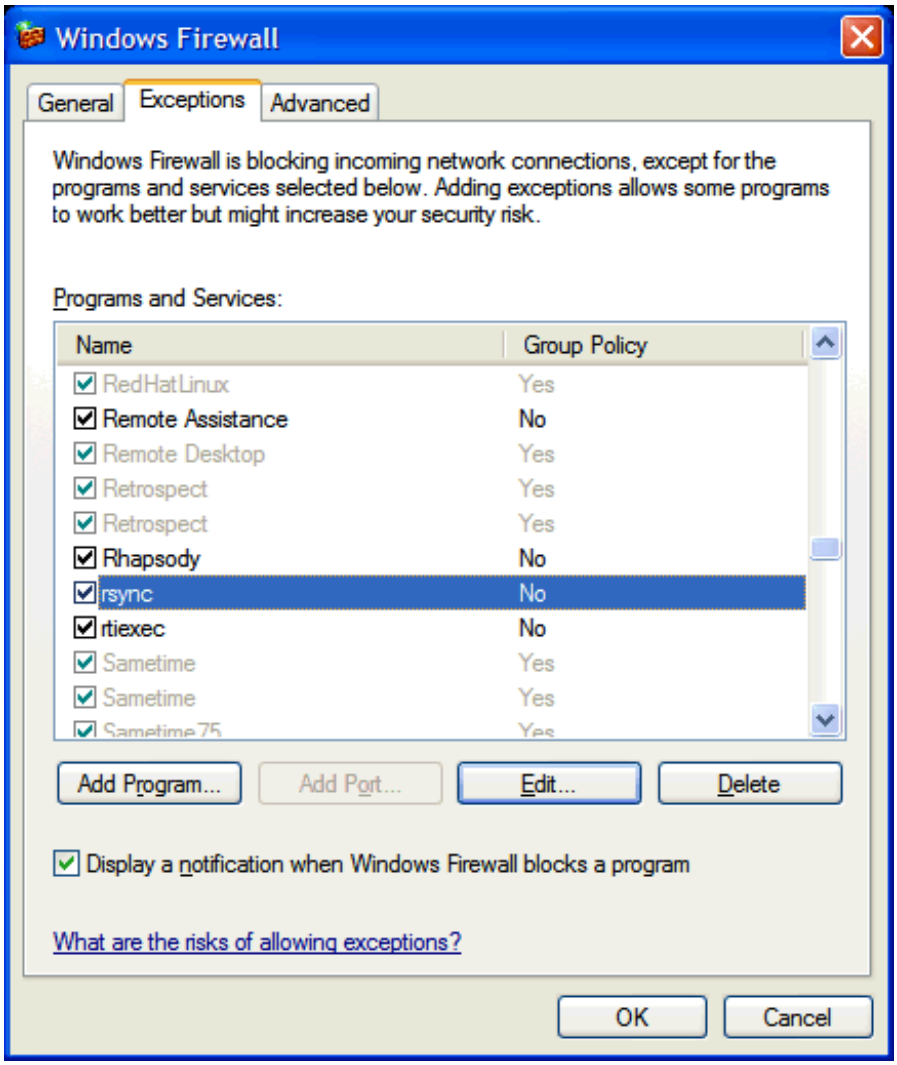
Find "rsync". If not running, select the service and select "start" or issue the command "`net start rsync`".

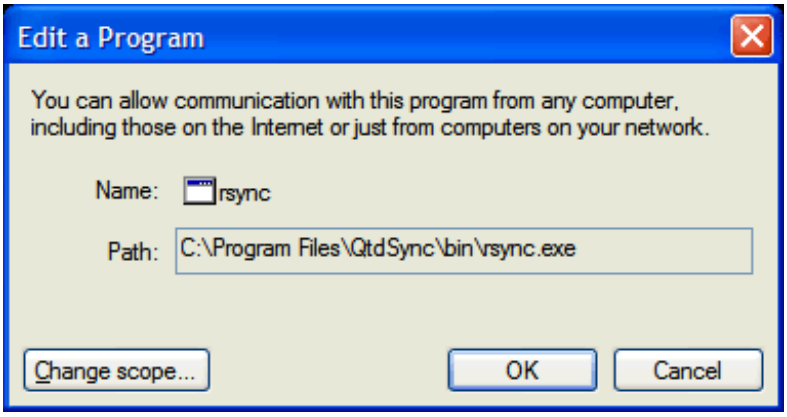
[Potential Pitfall]: You must check to see if the service has started. If not select " Start the service".

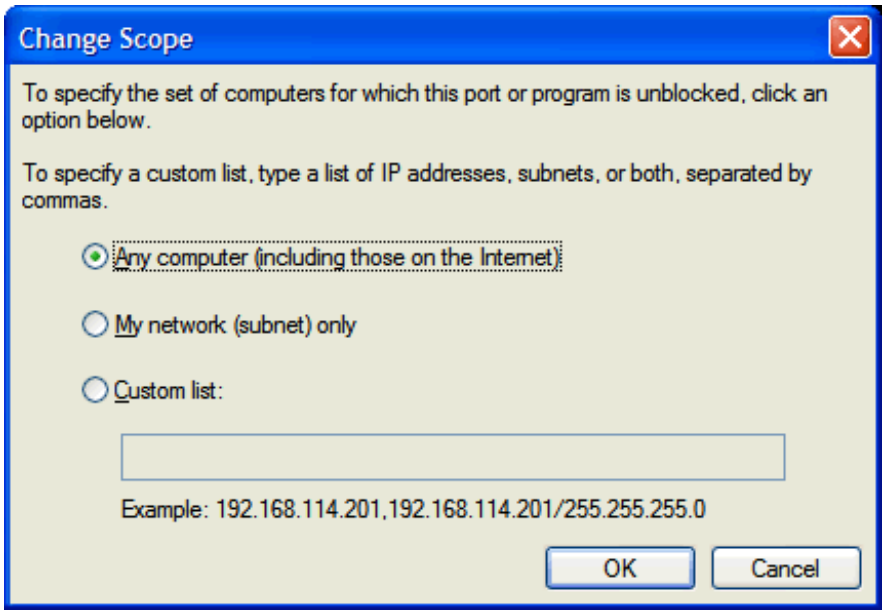
The Microsoft Windows firewall may not allow a system to attach to rsyncd on port 873. My experience is that the service is detected by the OS and added to the firewall privileges. I did not have to configure the firewall, registration was automatic. If the rsyncd service did not get registered with the MS/Windows firewall, one may have to specifically grant the permission to to the rsyncd service.

Select "Control panel" + Windows Firewall":


Windows Firewall
General
Exceptions
Advanced
For your security, some settings are controlled by Group Policy
Windows Firewall helps protect your computer by preventing unauthorized users from gaining access to your computer through the Internet or a network.
On (recommended)
This setting blocks all outside sources from connecting to this computer, with the exception of those selected on the Exceptions tab.
Don't allow exceptions
Select this when you connect to public networks in less secure locations, such as airports. You will not be notified when Windows Firewall blocks programs. Selections on the Exceptions tab will be ignored.
Off (not recommended)
Avoid using this setting. Turning off Windows Firewall may make this computer more vulnerable to viruses and intruders.
Windows Firewall is using your domain settings.
What else should I know about Windows Firewall?
OK
Cancel


Windows Firewall
General
Exceptions
Advanced
Windows Firewall is blocking incoming network connections, except for the programs and services selected below. Adding exceptions allows some programs to work better but might increase your security risk.
Programs and Services:
Name
Group Policy
RedHatLinux Yes
Remote Assistance No
Remote Desktop Yes
Retrospect Yes
Retrospect Yes
Rhapsody No
rsync No
rtiexec No
Sametime Yes
Sametime Yes
Sametime75 Yes
Add Program...
Add Port...
Edit...
Delete
Display a notification when Windows Firewall blocks a program
What are the risks of allowing exceptions?
OK
Cancel


Edit a Program
You can allow communication with this program from any computer, including those on the Internet or just from computers on your network.
Name: rsync
Path: C:\Program Files\QtdSync\bin\rsync.exe
Change scope...
OK
Cancel



To remove rsyncd from the available list of services, execute the following command from a "cmd" terminal:

```
c:\cygwin\bin\cygrunsrv -R rsyncd
```

Debugging Rsync:

Using telnet to test an rsync connection:

One can use telnet to test connectivity with an Rsync server:

```
[bash]$ telnet 192.168.1.12  873
Trying 192.168.1.12...
Connected to 192.168.1.12.
Escape character is '^]'.
@RSYNCD: 30.0
^C
Connection closed by foreign
host.
```

This is the proper response when using telnet to connect to rsync.

This is representative of a telnet connection to a misconfigured rsync server or one which has not yet been started (nothing for the client to connect to):

```
[bash]$ telnet 192.168.1.13  873
Trying 192.168.1.13...
 telnet: connect to address 192.168.1.13: Connection timed
out
```

does not get past this and times out.

Rsync client responses:

An rsync server which has been shut down or was never started will cause the rsync client to behave as follows:

```
[bash]$ rsync -avr 192.168.1.12::Proj1 /home/user1/Test
rsync: failed to connect to 192.1.1.12: Connection timed out (110)
rsync error: error in socket IO (code 10) at clientserver.c(124)
[receiver=3.0.6]
```

Rsync logs:

Check the rsyncd log file: `/var/log/rsyncd.log`

```
2011/09/08 21:13:10 [1032] rsyncd version 2.6.6 starting, listening on port
873
2011/09/08 21:17:10 [3156] rsync error: received SIGUSR1 or SIGINT (code 20)
```

compared to normal logging of a client connection and file transfer:

```
2011/09/08 21:19:10 [2796] rsync on Proj1/ from 192.168.1.10
2011/09/08 21:19:10 [2796] sent 940 bytes  received 334 bytes  total size
97
```

Links:

- Samba Rsync home page
- Lsyncd - Unix/Linux only. Monitor drive for changes (inotify) and perform an rsync any time a change is made.

Public rsync servers:

- rsync.net
- s3rsync.com
